**Лабораторная работа №5**

**«Хеш-таблицы»**

**Задание №1.**

Измените программу hash.java (см. листинг 11.1) так, чтобы в ней использовалось квадратичное пробирование.

Листинг 11.1. Программа hash.java

*// hash.java*

*// Реализация хеш-таблицы с линейным пробированием*

*// Запуск программы: C:>java HashTableApp*

**import** java.io.\*;

*////////////////////////////////////////////////////////////////*

**class** DataItem

{ *// (Данных может быть и больше)*

**private** **int** iData; *// Данные (ключ)*

*//--------------------------------------------------------------*

**public** DataItem(**int** ii) *// Конструктор*

{ iData = ii; }

*//--------------------------------------------------------------*

**public** **int** getKey()

{ **return** iData; }

*//--------------------------------------------------------------*

} *// Конец класса DataItem*

*////////////////////////////////////////////////////////////////*

**class** HashTable

{

**private** DataItem[] hashArray; *// Массив ячеек хеш-таблицы*

**private** **int** arraySize;

*// -------------------------------------------------------------*

**public** HashTable(**int** size) *// Конструктор*

{

arraySize = size;

hashArray = **new** DataItem[arraySize];

nonItem = **new** DataItem(-1); *// Ключ удаленного элемента -1*

}

*// -------------------------------------------------------------*

**public** **void** displayTable()

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Table: ");

**for**(**int** j=0; j<arraySize; j++)

{

**if**(hashArray[j] != **null**)

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print(hashArray[j].getKey() + " ");

**else**

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("\*\* ");

}

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("");

}

*// -------------------------------------------------------------*

**public** **int** hashFunc(**int** key)

{

**return** key % arraySize; *// Хеш-функция*

}

*// -------------------------------------------------------------*

**public** **void** insert(DataItem item) *// Вставка элемента данных*

*// (Метод предполагает, что таблица не заполнена)*

{

**int** key = item.getKey(); *// Получение ключа*

**int** hashVal = hashFunc(key); *// Хеширование ключа*

*// Пока не будет найдена*

**while**(hashArray[hashVal] != **null** && *// пустая ячейка или -1,*

hashArray[hashVal].getKey() != -1)

{

++hashVal; *// Переход к следующей ячейке*

hashVal %= arraySize; *// При достижении конца таблицы*

} *// происходит возврат к началу*

hashArray[hashVal] = item; *// Вставка элемента*

}

*// -------------------------------------------------------------*

**public** DataItem delete(**int** key) *// Удаление элемента данных*

{

**int** hashVal = hashFunc(key); *// Хеширование ключа*

**while**(hashArray[hashVal] != **null**) *// Пока не будет найдена*

*// пустая ячейка*

{ *// Ключ найден?*

**if**(hashArray[hashVal].getKey() == key)

{

DataItem temp = hashArray[hashVal]; *// Временное сохранение*

hashArray[hashVal] = nonItem; *// Удаление элемента*

**return** temp; *// Метод возвращает элемент*

}

++hashVal; *// Переход к следующей ячейке*

hashVal %= arraySize; *// При достижении конца таблицы*

} *// происходит возврат к началу*

**return** **null**; *// Элемент не найден*

}

*// -------------------------------------------------------------*

**public** DataItem find(**int** key) *// Поиск элемента с заданным ключом*

*// (Метод предполагает, что таблица не заполнена)*

{

**int** hashVal = hashFunc(key); *// Хеширование ключа*

**while**(hashArray[hashVal] != **null**) *// Пока не будет найдена*

*// пустая ячейка*

{ *// Ключ найден?*

**if**(hashArray[hashVal].getKey() == key)

**return** hashArray[hashVal]; *// Да, вернуть элемент*

++hashVal; *// Переход к следующей ячейке*

hashVal %= arraySize; *// При достижении конца таблицы*

} *// происходит возврат к началу*

**return** **null**; *// Элемент не найден*

}

*// -------------------------------------------------------------*

} *// Конец класса HashTable*

*////////////////////////////////////////////////////////////////*

**class** HashTableApp

{

**public** **static** **void** main([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] args) **throws** [IOException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+ioexception)

{

DataItem aDataItem;

**int** aKey, size, n, keysPerCell;

*// Ввод размеров*

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter size of hash table: ");

size = getInt();

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter initial number of items: ");

n = getInt();

keysPerCell = 10;

*// Создание таблицы*

HashTable theHashTable = **new** HashTable(size);

**for**(**int** j=0; j<n; j++) *// Вставка данных*

{

aKey = (**int**)(java.lang.[Math](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+math).random() \*

keysPerCell \* size);

aDataItem = **new** DataItem(aKey);

theHashTable.insert(aDataItem);

}

**while**(**true**) *// Взаимодействие с пользователем*

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter first letter of ");

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("show, insert, delete, or find: ");

**char** choice = getChar();

**switch**(choice)

{

**case** 's':

theHashTable.displayTable();

**break**;

**case** 'i':

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter key value to insert: ");

aKey = getInt();

aDataItem = **new** DataItem(aKey);

theHashTable.insert(aDataItem);

**break**;

**case** 'd':

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter key value to delete: ");

aKey = getInt();

theHashTable.delete(aKey);

**break**;

**case** 'f':

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Enter key value to find: ");

aKey = getInt();

aDataItem = theHashTable.find(aKey);

**if**(aDataItem != **null**)

{

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Found " + aKey);

}

**else**

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Could not find " + aKey);

**break**;

**default**:

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("Invalid entry**\n**");

}

}

}

*//--------------------------------------------------------------*

**public** **static** [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) getString() **throws** [IOException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+ioexception)

{

[InputStreamReader](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+inputstreamreader) isr = **new** [InputStreamReader](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+inputstreamreader)([System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).in);

[BufferedReader](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+bufferedreader) br = **new** [BufferedReader](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+bufferedreader)(isr);

[String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) s = br.readLine();

**return** s;

}

*//--------------------------------------------------------------*

**public** **static** **char** getChar() **throws** [IOException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+ioexception)

{

[String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) s = getString();

**return** s.charAt(0);

}

*//-------------------------------------------------------------*

**public** **static** **int** getInt() **throws** [IOException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+ioexception)

{

[String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) s = getString();

**return** [Integer](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+integer).parseInt(s);

}

*//--------------------------------------------------------------*

} *// Конец класса HashTableApp*

*////////////////////////////////////////////////////////////////*

**Задание №2.**

Реализуйте хеш-таблицу с линейным пробированием, в которой хранятся строки. Вам понадобится хеш-функция, преобразующая строку в индекс. Предполагается, что строки записаны символами нижнего регистра, поэтому 26 символов английского алфавита должно быть достаточно.